Resolving Ambiguity in Embodied Instructions via Semantic Valency Conflict

Anonymous ACL submission

Abstract

Natural language instructions in human-robot interaction often contain subtle ambiguities that hinder reliable interpretation. These ambiguities arise when a single instruction can be interpreted in multiple ways, assigning conflicting semantic roles to objects, tools, or participants, potentially leading to execution failures. To address this, we propose Semantic Valency Conflict (SVC), a cognitively inspired, logit-free method for detecting ambiguity in robot-directed instructions. SVC identifies divergences in role assignments across alternative interpretations of a predicate, using large language models (LLMs) to infer context-sensitive semantic frames. Our method is model-agnostic and compatible with both open- and closed-source LLMs. SVC produces clear, structured outputs that highlight which parts of the instruction are ambiguous and indicate which predicate and its associated arguments lead to multiple or conflicting interpretations. We evaluate SVC on two datasets, AmbiK and Introspective Planning, and demonstrate that it shows strong and consistent performance in detecting subtle ambiguities in natural language instructions given to robots in safety, unambiguous, and preference-based scenarios.

1 Introduction

Natural language is inherently polysemous, making it challenging for large language models (LLMs) to accurately follow instructions (Heo et al., 2025), interpret textual descriptions (Singhal et al., 2024; Chuganskaya et al., 2023), and perform planning tasks (Hazra et al., 2024; Grigorev et al., 2024, 2025). Depending on the context, multiple valid actions may exist in a given situation, making it difficult to select the most appropriate one. These challenges are associated with two types of uncertainty: epistemic, arising from lack of knowledge or insufficient data, and aleatoric, caused by random and unpredictable variations in the environment (Shorinwa et al., 2024). Both types of uncertainty complicate action selection by introducing ambiguity in outcome evaluation and adaptation to new situations.

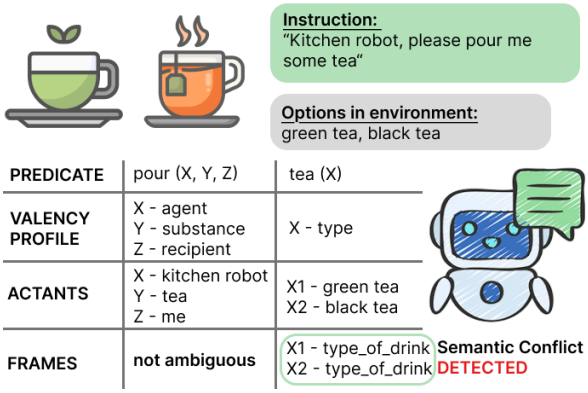

Figure 1: The robot receives the instruction "Pour me some tea" in a setting with green and black tea. The method detects ambiguity by identifying competing interpretations of the predicate "tea", where "green" and "black" act as conflicting actants within the same action frame, revealing a preference-related variability.

In robotics, one of the key challenges is following instructions given in natural language. Modern approaches to solving this problem primarily rely on LLMs, which, unlike heuristic methods, can flexibly interpret complex commands (Huang et al., 2022; Ahn et al., 2022; Kovalev and Panov, 2022; Sarkisyan et al., 2023; Onishchenko et al., 2025). However, user instructions can often be ambiguous due to factors such as the use of synonyms, metaphorical expressions, abbreviations, or environmental complexity where the same object may be represented in various forms (Fig. 1). This ambiguity, alongside known risks like LLM hallucinations, can lead to mission failures or safety hazards (Zhang et al., 2025). To mitigate these risks, it is essential to detect ambiguity promptly and clarify instructions. Implementing ambiguity detection plays a vital role in this process, ensuring the safe

integration of LLMs into intelligent agents (Firoozi et al., 2023). Accurate interpretation and execution of instructions are therefore crucial for building reliable and safe robotic systems.

Common techniques for ambiguity resolution include generating multiple candidate interpretations ranked by contextual relevance and interactive clarification (Ren et al., 2023). Some approaches also incorporate external knowledge bases to improve accuracy, but this reduces the system's autonomy and adaptability in novel environments (Liang et al., 2024). Although recent methods apply techniques like Conformal Prediction (CP) such as Su et al. (2024a) and affordance estimation Jr. and Manocha (2024) to improve ambiguity detection, they still struggle to effectively capture subtle ambiguities inherent in natural language instructions.

To address ambiguity in natural language instructions caused by lexical polysemy and underspecified argument structures, we propose the **Semantic Valency Conflict** (SVC) method. The key idea is that a single word (or lexeme) can trigger multiple, incompatible interpretations known as semantic frames, each expecting a different set of roles and participants to make sense in context (also called a valency profile, see Fig. 1). By modeling ambiguity as a conflict between these profiles under a given environment (set of objects, properties, and relations present in the external context), SVC determines whether an instruction supports multiple mutually exclusive interpretations.

The SVC method identifies lexical units whose frame-induced valency profiles are incompatible within the current environment. This is achieved through predicate identification, dynamic frame induction and contextual role alignment.

Unlike prior approaches, SVC detects fine-grained semantic conflicts without logit access or static knowledge bases, enabling interpretability and adaptability in both white- and black-box settings. Fine-grained ambiguity refers to subtle semantic inconsistencies arising at a detailed level of meaning, such as nuanced conflicts between semantic roles, argument structures, or contextual interpretations.

We evaluated SVC method ability to detect ambiguity on two fully textual datasets: IntroPlan Mobile Manipulation (Liang et al., 2024) and AmbiK (Ivanova et al., 2025). Compared to logit-based and heuristic baselines, which either fail to detect ambiguity or ignore uncertain cases entirely, SVC demonstrates superior performance in identifying semantically ambiguous instructions, particularly in user-preference contexts.

Statement of contributions. In this work, we propose Semantic Valency Conflict, a new method for ambiguity detection grounded in frame-based valency analysis. We develop logit-free architecture that is compatible with both white-box and black-box language models. Unlike prior work, our method with LLM dynamically induces cognitive frames without relying on static knowledge bases or plans, making it adaptable. Finally, SVC operates at the cognitive layer of instruction understanding, ensuring semantic reliability before planning. Its structured outputs can directly inform or condition the planner by providing disambiguated action frames, reducing misinterpretations that lead to execution errors.

2 Related Works

LLMs exhibit uncertainty due to external and internal factors. External factors, like data noise, ambiguity, and lack of contextual information can sometimes be mitigated with clarifying questions (Zhang and Choi, 2025). Internal factors, such as model architecture, training limitations, and probabilistic text generation can be reduced through model refinement or by increasing the number and diversity of in-context examples (Wang et al., 2025).

Various methods are used for resolving ambiguity, including token-based, self-verbalized, semantic-similarity, and mechanistic interpretability methods (Shorinwa et al., 2024). Model decision analysis using methods such as directed entailment graphs improves the transparency of LLM reasoning (Da et al., 2024). However, detecting uncertainty is only the first step. The key challenge is mitigating or resolving it.

One approach to reducing ambiguity in LLMs is the generation of clarifying questions, where the model requests additional context to improve confidence in its response (Zhang and Choi, 2025). Another strategy involves model ensembles, which aggregate outputs from multiple independently trained LLMs with different parameters and architectures, reducing overall uncertainty (Liu et al., 2024). Additionally, embeddings from small language models can be used to distinguish between different types of uncertainty (Ahdritz et al., 2024). Methods based on CP have also been proposed to calibrate uncertainty in LLM-based planning systems (Angelopoulos and Bates, 2022).

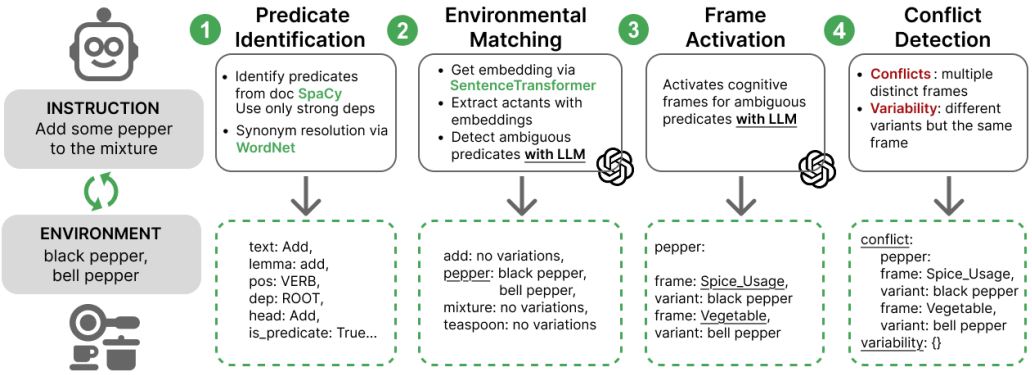

Figure 2: This figure illustrates the pipeline for detecting semantic ambiguity of the SVC method. The architecture decomposes the detection process into four main stages, each corresponding to a step in the cognitive mechanism of frame activation and conflict evaluation. The outputs of each stage of pipeline is illustrated using the example instruction: "Add some pepper to the mixture" and environment: black pepper, bell pepper. The system identifies potential ambiguity by tracing how the lexical unit pepper may evoke distinct semantic interpretations depending on context.

To address ambiguity resolving in robotic instruction-following, various approaches have been introduced in LLM-based planning. KnowNo Ren et al. (2023) is a framework that enables planners to assess and align uncertainty, helping them recognize when they lack confidence and need external input. This ensures statistically reliable task completion through CP. Building on this, Introspective Planning (IntroPlan) (Liang et al., 2024) integrates retrieval-augmented planning with CP, allowing models to proactively assess their confidence before taking action. By doing so, it reduces the number of user queries for task clarification. Another complementary approach is LAP (Jr. and Manocha, 2024), which includes the A-Feasibility metric. This metric combines scene context and model prompting to evaluate whether an action is both feasible and safe in environment. Ambiguities in large, shared spaces often arise from underspecified instructions that depend on implicit semantic features (e.g. cleanliness, fullness). To address this, Dogan et al. (2025) propose a model-agnostic approach leveraging iterative clarifications grounded in knowledge embeddings to infer missing attributes and improve object localization. A related line of work by Jiang et al. (2025) emphasizes that such vagueness frequently originates from referring expressions whose meaning is shaped by dialogue context and environmental factors.

In our work, we address a specific and underexplored source of ambiguity: semantic ambiguity in natural language instructions caused by lexical polysemy and underspecified argument roles. We treat ambiguity as a structural conflict between competing semantic frames activated by the same lexeme.

3 Background

In this section, we introduce key concepts used throughout the paper, **illustrative examples** are provided in Appendix A. Research in cognitive linguistics has shown that word meanings are shaped by underlying conceptual structures (Schwarze and Schepping, 1995; Bierwisch, 1983; Lehrer, 1990). Following frame semantics (Fillmore, 1985), we assume that understanding a lexeme requires reference to a conceptual **frame** specifying its semantic roles. Central to this view is the **lexical core** – the invariant meaning around which the lexeme's interpretations are structured.

We define the **semantic valency** of a lexeme L as the set of independent conceptual variables X necessary for interpreting L's core meaning. Lexemes that require one or more such variables are **predicates**, and the expressions that realize them in context are **semantic actants** (Testelets, 2001). Actants instantiate the predicate's roles in specific situations, linking conceptual meaning with linguistic realization. For instance, in the instruction "Pour me some tea", the predicate tea requires specification of a type (e.g., green or black). In a context where multiple instantiations of this actant are possible, such as green tea and black tea co-occurring in the environment, the valency slot remains underdetermined. Our method identifies this as a semantic conflict: multiple actants (types of tea) compete within the same frame (`Type_of_drink`), revealing ambiguity rooted in underspecified preference (see example in Fig. 1. Our ambiguity detection module follows Yarowsky (1993), who observed that

ambiguity often arises when a predicate activates incompatible valency structures simultaneously.

4 Method

Problem Formulation. Formally, let L be a lexeme (e.g. a verb, noun or pronoun), and let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of semantic valencies of L, where each x_i corresponds to an argument slot (actant) required for coherent interpretation. We define a **Semantic Valency Conflict** (SVC) as a case where multiple interpretations of L are simultaneously activated, and their corresponding valency profiles are incompatible with respect to the contextual environment $\mathcal{E}$. The environment $\mathcal{E}$ is the set of objects, properties, and relations present in the external context where the interpretation occurs. It provides the grounding necessary to resolve or exacerbate potential conflicts between valency profiles.

Each interpretation I_k of L activates a cognitive frame $\mathcal{F}_k$, which implicitly defines a *valency profile* $\mathcal{V}_k \subset \mathcal{D}$ – a set of expected participant roles and their semantic types. The domain $\mathcal{D}$ represents the space of all conceptually valid participant configurations for predicates in natural language.

To determine whether two valency profiles $\mathcal{V}_k$ and $\mathcal{V}_m$ are semantically compatible in a given environment $\mathcal{E}$, we use the relation $\sim_{\mathcal{E}}$. This relation denotes approximate contextual equivalence: $\mathcal{V}_k \sim_{\mathcal{E}} \mathcal{V}_m$ if their roles can be grounded to overlapping or compatible entities in $\mathcal{E}$, based on semantic similarity. In practice, this is operationalized using vector representations of candidate actants, their modifiers, and their types, matched against the entities and relations present in the environment.

If there exists $k \neq m$ such that $\mathcal{V}_k \not\sim_{\mathcal{E}} \mathcal{V}_m$, then L is ambiguous. For a detailed illustration of the valency conflict detection process, see Fig. 2.

It is important to note that in our approach the term predicate is used in a broad sense, encompassing not only verbs but also nouns or even entire phrases that activate distinct cognitive frames.

Overview of the SVC Method. To address the problem of detecting ambiguity arising from the lexical polysemy and underspecified argument structures, we propose the **Semantic Valency Conflict** method. It is based on the assumption that ambiguity in a natural language instruction is determined by a conflict between different semantic valencies of the same lexeme (Long et al., 2022; Apresjan, 2000). In other words, ambiguity arises when a single lexeme simultaneously activates multiple incompatible frames (cognitive scenarios) $\mathcal{F}$.

To implement the SVC method, we propose an architecture (see Fig. 2) where each stage addresses a specific task, from preprocessing to ambiguity detection. The overall architecture mirrors the cognitive process of frame activation and conflict resolution, and is implemented in four sequential stages. More details of each stage of SVC architecture and **algorithm** are provided in Appendix E.

The actants of each predicate are extracted and aligned with environmental constraints $\mathcal{E}$ (i.e., the set of relevant objects, properties, and relations in the instruction's context). The goal is to determine whether all participant roles required by the predicate are contextually supported. For each possible interpretation I_k of a predicate, the corresponding cognitive frame $\mathcal{F}_k$ is activated. Frame induction is performed dynamically using LLMs, allowing the system to infer plausible frames and their valency structures without relying on static knowledge bases (Liang et al., 2024; Chaplot and Salakhutdinov, 2018). Each frame implicitly defines a valency profile $\mathcal{V}_k$ over the domain $\mathcal{D}$.

At the final stage, the system evaluates the set of cognitive frames $\{\mathcal{F}_k\}$ activated for each predicate, each associated with a valency profile $\mathcal{V}_k$. The method compares valency profiles $\mathcal{V}_k$ for all competing interpretations. If there exist two interpretations $k \neq m$ such that their valency profiles are semantically incompatible with respect to the environment, formally expressed as

$$\mathcal{V}_k \not\sim_{\mathcal{E}} \mathcal{V}_m, \tag{1}$$

the predicate is flagged as ambiguous due to **frame conflict**. This condition indicates that the predicate simultaneously activates multiple, mutually exclusive event schemas, reflecting true semantic ambiguity – for example, the predicate *pepper* activating frames $\mathcal{F}_1 =$ `Spice_Usage` and $\mathcal{F}_2 =$ `Vegetable_Cooking`.

Alternatively, if only one frame $\mathcal{F}_k$ is active but multiple variants $\{v_i\}$ exist within this frame,

$$|\{v_i\}| > 1, \tag{2}$$

the system interprets this as a **choice within the scenario** rather than a semantic conflict. Such variability represents different referents or attributes within the same event schema (e.g., variants *olive oil* and *sunflower oil* under the frame `Cooking_Oil_Ingredient`). Although this does

not signify ambiguity at the frame level, it highlights distinctions that must be resolved for precise grounding or execution. Thus, the ambiguity detection mechanism differentiates between the cases providing a robust foundation for subsequent disambiguation steps.

5 Experimental Evaluation

To systematically assess the effectiveness and limitations of the proposed approach, we conduct a comprehensive experimental evaluation structured around the following research questions (RQs):

- **RQ1:** Can semantic conflict serve as a sufficient basis for ambiguity detection?
- **RQ2:** In which task types is semantic valency conflict most significant?
- **RQ3:** How do model size/type/architecture affect method performance?
- **RQ4:** Is simple prompting competitive?

Datasets. To evaluate SVC, we used two datasets: **IntroPlan Mobile Manipulation** (Liang et al., 2024) and **AmbiK** (Ivanova et al., 2025). Both contain robot instructions within a given environment and were used only for evaluation; no training or tuning was performed on them.

IntroPlan Mobile Manipulation dataset includes 600 short, single-step tasks with scene descriptions and object lists with the same distribution of different types of examples as in the KnowNo dataset (Ren et al., 2023). Although it contains ambiguous examples, it lacks precise ambiguity definitions and has an incomplete typology that overlaps with general linguistic phenomena.

AmbiK is a specialized dataset containing 2,000 annotated instructions covering three types of semantic ambiguity: user preferences, commonsense knowledge, and safety considerations. Each task has clarifying questions, answers, and execution plans for both ambiguous and unambiguous versions, and success markers for disambiguation.

Metrics. To evaluate the quality of ambiguity detection, we employ two binary metrics from Ivanova et al. (2025): **Help Rate (HR)** and **Correct Help Rate (CHR)**. **HR** measures how often the detection module determines that an instruction requires intervention – such as a clarifying question or a help request. The signal for this decision is the presence of either a semantic conflict between activated frames or the variability of possible actant interpretations within a single frame. **CHR** assesses the appropriateness of the help request decision, taking into account the actual ambiguity type annotated in the dataset. For unambiguous instructions, the robot should not request help; conversely, it should request help when ambiguity is present, depending on the type of instruction.

To assess the alignment between system behavior and user intent, we additionally report **Intent Alignment (IA)**, which captures whether the method's interpretation or suggested disambiguation correctly preserves the user's intended goal. IA is computed by embedding the set of ground-truth user intents and the set of interpretations produced by the method into a shared semantic space, and assigning a positive score if at least one predicted variant exceeds a cosine similarity threshold with ground-truth intent.

Finally, as a main metric, we introduce **Contextual Help Quality (CHQ)** to measure the usefulness of contextual clarifications in supporting user intent. CHQ evaluates the combined effect of accurate ambiguity detection and intent preservation, and is defined as the harmonic mean of IA and CHR:

$$\mathrm{CHQ} = \frac{2 \cdot \mathrm{IA} \cdot \mathrm{CHR}}{\mathrm{IA} + \mathrm{CHR} + \varepsilon}, \tag{3}$$

where ε is a small constant ($\varepsilon = 10^{-6}$) added for numerical stability. For more information on the metrics, see Appendix B.

Models and Baselines. We conducted experiments using **GPT-3.5-Turbo** (OpenAI, 2023)[1], **Mistral-7B-Instruct-v0.2** (MistralAI, 2023)[2], **Gemma-2-9B-IT** (Google, 2024)[3], and **Qwen1.5-7B-Chat** (QwenTeam, 2024)[4]. For brevity and clarity, we refer to these models simply as GPT-3.5, Mistral, Gemma, and Qwen, respectively. To compare the proposed SVC-based approach with a prompt-only baseline, we additionally employed the recently released **GPT-5-mini** model (OpenAI, 2025)[5]. For implementation details and prompts used see Appendix C.

We compare our method against **four baselines** previously proposed in the context of instruction

[1]Accessed via API: https://platform.openai.com

[2]Available at: https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

[3]Available at: https://huggingface.co/google/gemma-2-9b-it

[4]Available at: https://huggingface.co/Qwen/Qwen1.5-7B-Chat

[5]Accessed via API: https://platform.openai.com

Table 1: Performance across instruction types. For each model and metric, the best result is shown in **bold**, while the second-best result is underlined. Higher values are better for CHR, IA, and CHQ. Additional results with HR see in Appendix 4

Model	Method	Instruction type											
		Unambiguous			Common Sense			Preferences			Safety		
		CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑
GPT-3.5	LoFreeCP	0.77	0.18	0.29	0.80	0.10	0.17	0.15	0.10	0.12	0.24	0.10	0.14
	LAP	**1.00**	**0.41**	**0.58**	**1.00**	0.20	0.33	0.00	0.10	0.00	0.00	0.20	0.00
	KnowNo	**1.00**	0.35	0.51	**1.00**	0.21	**0.34**	0.00	0.10	0.00	0.00	0.22	0.00
	SVC	0.51	0.35	0.41	0.51	**0.26**	**0.34**	**0.77**	**0.58**	**0.66**	**0.43**	**0.26**	**0.32**
Mistral	LoFreeCP	0.28	**0.69**	**0.39**	0.31	0.40	**0.34**	0.73	0.52	0.60	**0.62**	**0.50**	**0.55**
	LAP	0.93	0.12	0.21	0.95	0.05	0.09	0.06	0.04	0.04	0.04	0.02	0.02
	KnowNo	**1.00**	0.02	0.03	**1.00**	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00
	SVC	0.51	0.32	**0.39**	0.23	**0.43**	0.30	**0.77**	**0.74**	**0.75**	0.44	0.45	0.44
Qwen	LoFreeCP	0.36	**0.24**	0.28	0.26	**0.28**	0.27	0.62	0.32	0.42	0.17	0.23	0.19
	LAP	0.73	0.22	**0.33**	0.64	0.21	0.31	0.28	0.17	0.21	0.29	0.17	0.21
	KnowNo	**1.00**	0.08	0.14	**1.00**	0.02	0.03	0.00	0.19	0.00	0.00	0.00	0.00
	SVC	0.54	0.17	0.25	0.61	0.27	**0.37**	**0.71**	**0.37**	**0.48**	**0.39**	**0.25**	**0.30**
Gemma	LoFreeCP	0.51	**0.41**	**0.45**	0.44	**0.20**	0.27	0.20	0.24	0.21	**0.34**	0.09	0.14
	LAP	0.66	0.12	0.20	0.89	0.04	0.07	0.00	0.10	0.00	0.13	0.05	0.07
	KnowNo	**1.00**	0.09	0.16	**1.00**	0.00	0.00	0.00	0.17	0.00	0.00	0.00	0.00
	SVC	0.83	0.26	0.39	0.84	0.17	**0.28**	**0.37**	**0.54**	**0.43**	0.21	**0.32**	**0.25**

following and embodied agents. **KnowNo** (Ren et al., 2023) is one of the first methods to apply conformal prediction for ambiguity detection in kitchen task scenarios with LLMs. The model is prompted to generate multiple candidate interpretations and then select the most appropriate one. **LAP** (Jr. and Manocha, 2024) extends KnowNo by incorporating affordance estimation. For each candidate response, it combines the model confidence with two affordance scores, Context-Based Affordance and Prompt-Based Affordance **LoFreeCP** (Su et al., 2024b) avoids logit access and instead applies uncertainty-based conformal prediction over multiple LLM generations. In addition, we include a **Prompting** baseline, representing a standard instruction-based ambiguity detection setup without conformal prediction or external supervision. The LLM is prompted to classify an instruction as Ambiguous or Unambiguous given the environment and, if ambiguous, to produce a minimal set of alternative interpretations (see Appendix C). We consolidate the **Binary** and **NoHelp** baselines from (Ren et al., 2023) into this unified **Prompting** baseline, as both rely on direct prompting and differ only in whether uncertainty is explicitly expressed: **Binary** labels its output as Certain or Uncertain and requests clarification accordingly, whereas **NoHelp** always outputs a single answer without requesting clarification.

Results. Our experimental evaluation demonstrates that the proposed Semantic Valency Conflict method provides a robust, semantically grounded mechanism for detecting ambiguity in embodied instructions. Across both datasets, SVC competitive with many established baselines, with particularly strong gains on instruction types related to safety-critical scenarios and unambiguous instructions, where precise predicate grounding is essential.

RQ1: Semantic conflict is sufficient for ambiguity detection. The strong performance of SVC, particularly on safety-based and contextually underspecified instructions, empirically validates that conflict between induced semantic frames is a sufficient and robust signal. The AmbiK dataset (Table 1) reveals SVC's principal strength: identifying ambiguity arising from user preference. In this category, SVC achieves a CHR of 0.77 (vs. 0.15 for LoFreeCP on GPT and 0.73 on Mistral).

RQ2: Semantic valency conflict is most significant in underspecified instruction types. It is most critical for Underspecification tasks. These tasks involve a predicate whose actants can be realized by multiple distinct entities in the environment. The conflict emerges from the choice between these realizations. Table 2 shows that SVC

Table 2: Results on the IntroPlan dataset across instruction types and models. The table reports CHR, IA, and CHQ for Prompting, LAP, LoFreeCP, and SVC. For each model and metric, the best result is shown in **bold**, while the second-best result is underlined. The KnowNo baseline was excluded from this table because preliminary evaluation showed it consistently produced degenerate outputs (HR = 0) by failing to identify ambiguous instructions.

	Prompting			LAP			LoFreeCP			SVC		
Instruction Type	CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑	CHR↑	IA↑	CHQ↑
Mistral-7B-Instruct												
creative multilabel	0.00	0.77	0.00	0.19	0.22	0.20	1.00	0.00	0.00	0.22	1.00	**0.36**
single-label	1.00	0.68	**0.81**	0.80	0.25	0.38	0.00	0.00	0.00	0.40	0.95	0.56
winograd	1.00	0.73	**0.84**	0.30	0.24	0.27	1.00	0.13	0.23	0.00	0.00	0.00
multi-label	0.00	0.73	0.00	0.57	0.42	0.48	1.00	0.00	0.00	0.38	0.88	**0.53**
unambiguous	1.00	0.84	0.91	0.92	0.13	0.23	0.00	0.00	0.00	0.98	1.00	**0.99**
spatial amb	0.00	0.95	0.00	0.00	0.01	0.00	1.00	0.00	0.00	0.13	1.00	**0.23**
unsafe	0.00	0.67	0.00	0.08	0.17	0.11	1.00	0.00	0.00	0.44	1.00	**0.61**
creative single-label	1.00	0.79	**0.88**	0.73	0.31	0.44	0.00	0.00	0.00	0.69	0.50	0.58
Qwen1.5-7B-Chat												
creative multi-label	0.09	0.58	0.16	0.21	0.31	0.25	1.00	0.17	0.29	0.23	0.84	**0.36**
single-label	0.17	0.82	0.28	0.64	0.29	0.40	0.00	0.00	0.00	0.59	0.51	**0.55**
winograd	1.00	0.81	**0.90**	0.47	0.22	0.30	1.00	0.06	0.11	0.00	0.00	0.00
multi-label	0.00	0.84	0.00	0.63	0.53	**0.58**	1.00	0.00	0.00	0.27	0.86	0.41
unambiguous	0.51	0.35	0.42	0.84	0.15	0.25	0.00	0.00	0.00	0.34	0.72	**0.46**
spatial amb	0.16	0.47	0.24	0.00	0.04	0.00	1.00	0.00	0.00	0.33	0.21	**0.26**
unsafe	0.00	0.68	0.00	0.12	0.20	0.15	1.00	0.00	0.00	0.64	0.73	**0.68**
creative single-label	0.17	0.82	0.28	0.62	0.38	**0.47**	0.00	0.00	0.00	0.72	0.13	0.22
Gemma-2-9B-IT												
creative multi-label	0.00	0.00	0.00	0.14	0.28	0.19	1.00	0.14	0.25	0.23	0.84	**0.36**
single-label	1.00	0.13	0.23	0.56	0.22	0.32	0.00	0.00	0.00	0.59	0.51	**0.55**
winograd	0.00	0.00	0.00	0.34	0.22	**0.27**	1.00	0.00	0.00	0.00	0.00	0.00
multi-label	0.00	0.31	0.00	0.49	0.62	**0.55**	1.00	0.12	0.21	0.27	0.86	0.41
unambiguous	1.00	0.04	0.08	0.89	0.15	0.26	0.00	0.00	0.00	0.34	0.72	**0.46**
spatial amb	0.00	0.00	0.00	0.00	0.02	0.00	1.00	0.00	0.00	0.33	0.21	**0.26**
unsafe	0.00	0.00	0.00	0.10	0.19	0.13	1.00	0.04	0.08	0.64	0.73	**0.68**
creative single-label	1.00	0.00	0.00	0.76	0.34	**0.47**	0.00	0.00	0.00	0.72	0.13	0.22

outperforms almost all baselines on unambiguous and safety-related (unsafe) instruction types across all evaluated models. In unambiguous settings, SVC achieves near-perfect CHR and CHQ scores (e.g., up to 0.99 on Mistral-7B), indicating reliable rejection of false ambiguity, whereas prompting-based and uncertainty-driven baselines frequently over-trigger ambiguity signals. In safety-critical tasks, SVC yields substantially higher CHQ and IA scores, reflecting more accurate identification of ambiguity, while baselines miss such cases or collapse to degenerate behavior (CHR = 0).

RQ3: SVC generalizes across architectures. Performance is consistent across model families, but quality is influenced by instruction-tuning. SVC's logit-free design makes it portable. As seen in Table 1, it works effectively with GPT, Mistral, Qwen, and Gemma. However, the quality of the induced frames and consequently the IA scores tends to be higher with instruction-tuned models (Mistral-7B-Instruct and Gemma-2-9B-IT). These models produce more structured, deterministic frame descriptions, which improves the downstream matching and conflict analysis.

RQ4: Simple prompting is less effective than SVC. While simple prompting underperforms SVC on structural ambiguity metrics (e.g., CHQ), preference-based results are comparable (see Fig. 3). This is expected, as preference judgments rely primarily on surface-level plausibility rather than explicit decomposition of semantic conflicts. In such settings, strong models can approximate the correct preference without isolating the underlying semantic conflict, reducing the observ-

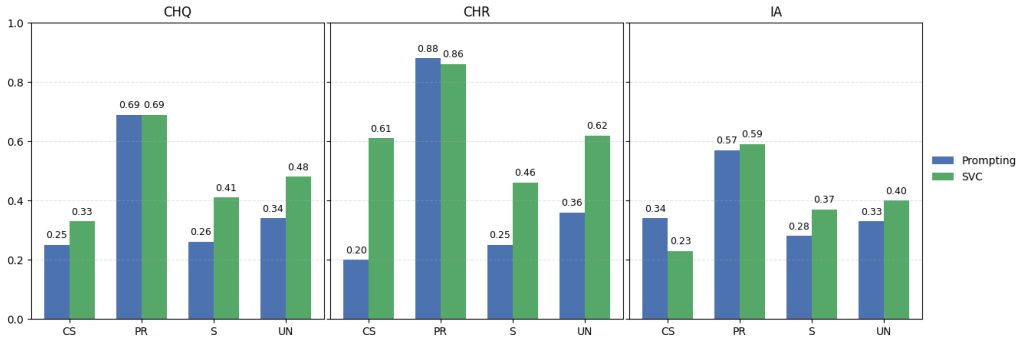

Figure 3: Comparison of CHQ performance between **Prompting** (prompt-only LLM baseline) and the proposed **SVC** method on the AmbiK dataset. Across all instruction types (Unambiguous (UN), Preference (PR), Common Sense (CS), and Safety (S)), SVC outperforms direct prompting in majority of cases, indicating improved robustness in ambiguity detection. The experiment is conducted using the modern **GPT-5-mini** model to reflect the current capabilities of large language models in handling complex ambiguities in natural language instructions.

Table 3: CHQ on a 150-sample AmbiK subset across all instruction types (Unambiguous (UN), Preference (PR), Common Sense (CS), and Safety (S)) using Mistral-7B. Best mean values are in **bold**. Abbreviations: P_I – predicate identification, S_E – synonym expansion, M_H – modifier handling, E_M – environment matching.

Ablation	CS	PR	S	UN
baseline	0.13±0.02	**0.60±0.03**	**0.41±0.04**	**0.43±0.02**
w/o P_I	0.13±0.02	0.58±0.04	0.39±0.05	**0.43±0.03**
w/o S_E	0.14±0.03	0.53±0.05	0.31±0.04	0.30±0.04
w/o M_H	0.06±0.02	0.58±0.04	0.31±0.03	0.26±0.03
w/o E_M	**0.30±0.04**	0.55±0.03	0.40±0.04	0.11±0.02

able gap between prompting and SVC.

6 Ablation Study

To further quantify the contribution of each module, we conducted systematic ablation studies to assess model sensitivity and component-level importance (see Table 3). The evaluation was performed on a 150-sample subset of the AmbiK dataset using the **Mistral-7B** model. To account for stochastic variability in LLM outputs, each ablation setting was evaluated over five independent runs with different random seeds, and the reported results are presented as the mean performance.

Pipeline exhibits strong robustness: removing most modules results in only moderate performance degradation. The most pronounced drop is observed when **environment matching** (E_M) is removed. This leads to a sharp decline in Unambiguous (UN) performance (0.43 to 0.11), highlighting the critical role of explicit grounding in the environment for resolving referential and underspecification ambiguities. Interestingly, the same ablation yields an apparent improvement on Common Sense (CS) tasks (0.13 to 0.30). This suggests that CS ambiguities are primarily resolved at the semantic or conceptual level and do not require explicit environment grounding. In such cases, environment matching may introduce spurious constraints, obscuring otherwise straightforward interpretations. **Synonym expansion** (S_E) and **modifier handling** (M_H) contribute consistent but secondary gains, particularly for CS and Safety tasks, supporting the importance of lexical and compositional refinement. In contrast, removing **predicate identification** (P_I) results in minimal variation across all instruction types, indicating that the induced frame structure is largely stable once a predicate is identified.

These results confirm that SVC benefits from modular grounding mechanisms when task structure requires them, while remaining robust and in some cases more effective when such grounding is unnecessary.

7 Conclusion

In this work, we introduce **Semantic Valency Conflict**, a cognitively inspired method for detecting ambiguity in natural language instructions. The approach analyzes conflicts between semantic valency profiles generated by contextually activated cognitive frames. Method outperforms baselines on unambiguous and safety-related instruction types across all evaluated models, effectively identifies subtle forms of ambiguity. SVC requires neither access to model logits nor reliance on static knowledge bases, ensuring compatibility with both white-box and black-box systems while maintaining adaptability to novel environments and tasks. Moreover, SVC's structured output provides interpretable diagnostics of ambiguity sources, facilitating effective downstream clarification and resolution mechanisms.

8 Limitations

Despite its effectiveness in detecting semantic valency conflicts, the current implementation of the SVC method is subject to several limitations.

Environment completeness dependency. The accuracy of ambiguity detection is highly contingent on the completeness of the environment model. If critical entities, properties, or relations are missing from $\mathcal{E}$, the system may fail to identify conflicts. Thus, noisy environment representation can undermine the method's reliability.

Restriction to prototypical valencies. Our approach focuses on core, prototypical valency structures, excluding non-canonical and metaphorical usages. As a result, polysemous extensions – such as metaphorical interpretations or constructions with optional actants – are not captured by the current model and may lead to misclassification or under-detection of ambiguity. For example, the instruction "Kill the heat" is a metaphorical expression meaning turn off the stove or reduce the flame, but the model may misinterpret it due to the lack of a literal, manipulable object. Similarly, in "Season the soup", the core valency structure (verb "season" + object "soup") omits the specific substance to be added (e.g., salt, pepper, herbs), which is crucial for task execution but remains implicit and therefore may not be flagged as ambiguous.

Syntactic and pragmatic ambiguity. SVC's performance collapses on Winograd tasks (CHR=0). These tasks hinge on coreference resolution and syntactic parsing. Since SVC operates on predicate-argument semantics and lacks a deep syntactic or discourse model, it cannot detect ambiguities where the semantic roles are stable but their referents are not.

LLM dependency. LLMs are used in multiple stages of the pipeline enhances the adaptability of the system compared to reliance on static knowledge bases, it introduces a trade-off between flexibility and precision. Specifically, LLMs are employed during frame activation to propose potential interpretations for ambiguous actions, and as a fallback mechanism to assess the ambiguity of actants whose reference to the environment remains uncertain. However, the primary actant extraction and entity alignment rely on symbolic parsing and embedding-based similarity, thereby reducing both computational load and epistemic dependency on the LLM. This hybrid design strikes a balance between the adaptability offered by LLMs and the robustness of structured, similarity-driven matching. Nevertheless, the overall reliability of ambiguity detection still partially depends on the quality of LLM-generated outputs at these two stages, which may introduce noise or contextually inappropriate variants.

Single-turn instruction processing. The current system operates on isolated, single-turn instructions paired with a static environment description. It does not support multi-turn dialogue or interactive clarification.

9 Ethical Considerations

Our approach uses LLMs in generation mode. Even when using a prompt that limits the output format, the model may still generate inappropriate and/or offensive content. In addition, LLMs are prone to hallucinations and can produce unexpected results. Therefore, giving them control over potentially harmful machines, such as robots, and testing such machines should be done in a regulated manner in a specially designated area with limited access for the people involved in the experiments. It is also possible to deliberately instruct a robot to execute harmful actions.

References

Gustaf Ahdritz, Tian Qin, Nikhil Vyas, Boaz Barak, and Benjamin L. Edelman. 2024. Distinguishing the knowable from the unknowable with language models. *Computing Research Repository*, arXiv:2402.03563. Version 2.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, and 1 others. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Anastasios N. Angelopoulos and Stephen Bates. 2022. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *Computing Research Repository*, arXiv:2107.07511.

Jurij D. Apresjan. 2000. Regular polysemy and lexical functions. In Igor Mel'čuk and Leo Wanner, editors, *Systematic Lexicography*, pages 119–132. Oxford University Press, Oxford.

Manfred Bierwisch. 1983. Semantische und conceptuelle repräsentation lexikalischer einheiten. In *Untersuchungen zur Semantik*. Akademie Verlag.

Devendra Singh Chaplot and Ruslan Salakhutdinov. 2018. Knowledge-based word sense disambiguation using topic models. *Computing Research Repository*, arXiv:1801.01900.

Anfisa A Chuganskaya, Alexey K Kovalev, and Aleksandr Panov. 2023. The problem of concept learning and goals of reasoning in large language models. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 661–672. Springer.

Longchao Da, Tiejin Chen, Lu Cheng, and Hua Wei. 2024. Llm uncertainty quantification through directional entailment graph and claim level response augmentation. *Computing Research Repository*, arXiv:2407.00994. Version 2.

Fethiye Irmak Dogan, Maithili Patel, Weiyu Liu, Iolanda Leite, and Sonia Chernova. 2025. A model-agnostic approach for semantically driven disambiguation in human-robot interaction. *Preprint*, arXiv:2409.17004.

Charles J. Fillmore. 1985. Frames and the semantics of understanding. *Quaderni di Semantica*, 6(2):222–254.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. 2023. Foundation models in robotics: Applications, challenges, and the future. *Computing Research Repository*, arXiv:2312.07843.

Google. 2024. Gemma: Open models based on gemini research. Technical report.

Danil Grigorev, Alexey Kovalev, and Aleksandr Panov. 2025. Verifyllm: Llm-based pre-execution task plan verification for robots. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Danil S Grigorev, Alexey K Kovalev, and Aleksandr I Panov. 2024. Common sense plan verification with large language models. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 224–236. Springer.

Rishi Hazra, Pedro Zuidberg Dos Martires, and Luc De Raedt. 2024. Saycanpay: Heuristic planning with large language models using learnable domain knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20123–20133.

Juyeon Heo, Miao Xiong, Christina Heinze-Deml, and Jaya Narain. 2025. Do llms estimate uncertainty well in instruction-following? In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Anastasia Ivanova, Bakaeva Eva, Zoya Volovikova, Alexey Kovalev, and Aleksandr Panov. 2025. AmbiK: Dataset of ambiguous tasks in kitchen environment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33216–33241, Vienna, Austria. Association for Computational Linguistics.

Chenxi Jiang, Chuhao Zhou, and Jianfei Yang. 2025. Rei-bench: Can embodied agents understand vague human instructions in task planning? *Preprint*, arXiv:2505.10872.

James F. Mullen Jr. and Dinesh Manocha. 2024. Lap, using action feasibility for improved uncertainty alignment of large language model planners. *Computing Research Repository*, arXiv:2403.13198.

Aleksei Konstantinovich Kovalev and Aleksandr Igorevich Panov. 2022. Application of pretrained large language models in embodied artificial intelligence. In *Doklady Mathematics*, volume 106, pages S85–S90. Springer.

Adrienne Lehrer. 1990. Polysemy, conventionality and the structure of the lexicon. *Cognitive Linguistics*.

Kaiqu Liang, Zixu Zhang, and Jaime Fernández Fisac. 2024. Introspective planning: Aligning robots' uncertainty with inherent task ambiguity. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*.

Linyu Liu, Yu Pan, Xiaocheng Li, and Guanting Chen. 2024. Uncertainty estimation and quantification for llms: A simple supervised approach. *Computing Research Repository*, arXiv:2404.15993.

Chen Long, Laura Kallmeyer, and Rainer Osswald. 2022. A frame-based model of inherent polysemy, copredication and argument coercion. In *Proceedings of the Workshop on Cognitive Aspects of the Lexicon*, pages 58–67, Taipei, Taiwan. Association for Computational Linguistics.

MistralAI. 2023. Mistral-7b-instruct-v0.2. `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2`. Accessed: 2025-08-01.

Anatoly Onishchenko, Alexey Kovalev, and Aleksandr Panov. 2025. Lookplangraph: Embodied instruction following method with vlm graph augmentation. In *Workshop on Reasoning and Planning for Large Language Models*.

OpenAI. 2023. Gpt-3.5-turbo (august 16 version). `https://openai.com`. Accessed: 2024-08-16.

OpenAI. 2025. Gpt-5 technical report. Accessed via OpenAI API.

QwenTeam. 2024. Qwen1.5-7b-chat. https://huggingface.co/Qwen/Qwen1.5-7B-Chat. Accessed: 2025-08-01.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. 2023. Robots that ask for help: Uncertainty alignment for large language model planners. In *Proceedings of the Conference on Robot Learning (CoRL)*.

Christina Sarkisyan, Alexandr Korchemnyi, Alexey K Kovalev, and Aleksandr I Panov. 2023. Evaluation of pretrained large language models in embodied planning tasks. In *International Conference on Artificial General Intelligence*, pages 222–232. Springer.

Christa Schwarze and Marie-Theres Schepping. 1995. Polysemy in a two-level-semantics. In *Current Issues in Linguistic Theory: Lexical Knowledge in the Organization of Language*. John Benjamins Publishing Company.

Ola Shorinwa, Zhiting Mei, Justin Lidard, Allen Z. Ren, and Anirudha Majumdar. 2024. A survey on uncertainty quantification of large language models: Taxonomy, open research challenges, and future directions. *Computing Research Repository*, arXiv:2412.05563.

Anmol Singhal, Chirag Jain, Preethu Rose Anish, Arkajyoti Chakraborty, and Smita Ghaisas. 2024. Generating clarification questions for disambiguating contracts. *Computing Research Repository*, arXiv:2403.08053.

Jiayuan Su, Jing Luo, Hongwei Wang, and Lu Cheng. 2024a. Api is enough: Conformal prediction for large language models without logit-access. *Preprint*, arXiv:2403.01216.

Jiayuan Su, Jing Luo, Hongwei Wang, and Lu Cheng. 2024b. Api is enough: Conformal prediction for large language models without logit-access. *Preprint*, arXiv:2403.01216.

Yakov G. Testelets. 2001. *Introduction to General Syntax*. Russian State University for the Humanities, Moscow.

Yifei Wang, Yu Sheng, Linjing Li, and Daniel Zeng. 2025. Uncertainty unveiled: Can exposure to more in-context examples mitigate uncertainty for large language models? *Preprint*, arXiv:2505.21003.

David Yarowsky. 1993. One sense per collocation. In *Proceedings of the Workshop on Human Language Technology*.

Hangtao Zhang, Chenyu Zhu, Xianlong Wang, Ziqi Zhou, Changgan Yin, Minghui Li, Lulu Xue, Yichen Wang, Shengshan Hu, Aishan Liu, Peijin Guo, and Leo Yu Zhang. 2025. Badrobot: Jailbreaking embodied LLMs in the physical world. In *The Thirteenth International Conference on Learning Representations*.

Michael J. Q. Zhang and Eunsol Choi. 2025. Clarify when necessary: Resolving ambiguity through interaction with lms. In *Findings of the 2025 North American Chapter of the Association for Computational Linguistics (NAACL)*.

A Appendix A – Glossary of Terms

- **Frame (Semantic Frame / Cognitive Frame)** A structured conceptual schema that organizes knowledge about a specific type of situation, event, or object, including the roles and relationships involved. Frames enable humans to interpret word meanings by invoking entire cognitive scenarios.
  Example: The sentence "The seller gave the customer some apples for five dollars" activates a COMMERCE frame involving roles such as SELLER, BUYER, GOODS, MONEY, and TRANSFER ACTION.

- **Cognitive Frame** Emphasizes the mental and experiential nature of a frame as a structure in the speaker's conceptual system.
  Example: Interpreting the word "pepper" involves activating a mental model of either a cooking spice or a vegetable, depending on context.

- **Semantic Frame** Highlights the lexical-semantic dimension of frames, focusing on how word meanings and their valency structures relate to structured scenarios.
  Example: "Black pepper" activates the SPICE_USAGE frame; "bell pepper" activates the VEGETABLE_COOKING frame.

- **Cognitive Scenario** A dynamic instantiation of a frame, often used synonymously, placing emphasis on the sequence of events and interactions among participants.
  Example: A request like "pour me some tea" evokes a scenario involving serving, a recipient, and a beverage.

- **Predicate** A lexeme that requires one or more participants to complete its meaning, forming the semantic core around which arguments are organized.
  Example: In "pour me some tea," the verb *pour* functions as a predicate that expects arguments like a recipient and a substance.

- **Lexical Core** The central, invariant meaning of a polysemous lexeme, abstracted from contextual variation. It provides the conceptual foundation from which semantic valencies emerge.
  Example: The noun *bank* has different senses ("financial institution" vs. "riverbank"), but in the financial sense, the lexical core refers to an institution that holds and manages money, regardless of specific services or context.

- **Semantic Valency** The set of argument slots that a lexeme (typically a predicate) conceptually requires in order to be fully interpreted. These slots correspond to independent semantic variables.
  Example: In the utterance "pour me some tea," the predicate *tea* has a unary valency slot for *type of tea*, filled by either *green* or *black*.

- **Valency** An expression in a sentence that fills one of the semantic valency slots of a predicate.
  Example: Green and *black* are arguments for the valency slot in *tea* denoting type.

- **Semantic Actant** A linguistic expression that realizes one of the predicate's semantic roles in a sentence. Actants serve as syntactic and semantic instantiations of frame roles.
  Example: In "The robot gave the customer tea," the expressions *robot*, *customer*, and *tea* are actants corresponding to AGENT, RECIPIENT, and THEME roles, respectively.

B Appendix B – Metrics Details

Intent Alignment (IA) evaluates whether the clarification request issued by the model is *useful*, i.e., whether it exposes alternative interpretations that are aligned with the user's underlying intent annotation. IA is a binary proxy metric and does not aim to measure full intent recovery, but rather whether at least one generated clarification aligns with the annotated user intent.

IA is computed only for instructions where the model requested help (HR $= 1$). For all other cases, IA is undefined and excluded from aggregation.

Algorithm 1: Intent Alignment (IA)

Input: User intent annotation I, generated variants V, embedding model, threshold τ
Output: IA $\in \{0, 1\}$
if $I = \emptyset$ **then**
  return NaN
else
  Encode I and V using the embedding model;
  Compute cosine similarities S_{ij} for all $i \in I, j \in V$;
  $s_{\max} \leftarrow \max_{i,j} S_{ij}$;
  if $s_{\max} \geq \tau$ **then**
    return 1
  else
    return 0

Help Rate (HR) HR measures how often the detection module determines that an instruction requires intervention – such as a clarifying question or a help request. The signal for this decision is the presence of either a *semantic conflict* between activated frames or *variability of possible actant interpretations* within a single frame. Formally,

$$\text{HR} = \begin{cases} 1, & \text{if detected ambiguous,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Correct Help Rate (CHR) CHR assesses the appropriateness of the help request decision, taking into account the actual ambiguity type annotated in the dataset.

For unambiguous instructions, the robot should not request help; conversely, it should request help when ambiguity is present. Thus, CHR measures the *selectivity and accuracy* of the detection module. It is defined as follows depending on the ambiguity type.

The **AmbiK dataset** contains the following instruction types: *user preferences*,*common sense knowledge*, *safety* and *unambiguous tasks*.

- For tasks of type *user preferences* and *safety*, the model is considered correct if it requested help:

$$\text{CHR} = \begin{cases} 1, & \text{if HR} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

- For other types (*common sense knowledge*, as well as *unambiguous tasks*), the model should succeed without requesting help:

$$\text{CHR} = \begin{cases} 1, & \text{if HR} = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

The **IntroPlan dataset** contains the following instruction types: *creative_multilabel_task*, *singlelabel_task*, *winograd_task*, *multilabel_task*, *unambiguous_task*, *spatial_ambiguous_task*, *unsafe_task*, and *creative_singlelabel_task*.

- We consider as ambiguous *creative_multilabel_task*, *winograd_task*, *multilabel_task*, *unsafe_task* and *spatial_ambiguous_task*, instructions where clarifications are generally expected

$$\text{CHR} = \begin{cases} 1, & \text{if HR} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

- For other types (*unambiguous_task*, *singlelabel_task*, *creative_singlelabel_task*), the model should succeed without requesting help:

$$\text{CHR} = \begin{cases} 1, & \text{if HR} = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

This differentiated definition of CHR allows the evaluation to account for the specific nature of various ambiguity types: clarifications are necessary for user preference ambiguities, while for tasks relying on common knowledge or obvious actions, a help request may be redundant.

C Appendix C – Implementation Experimental Setup

C.1 Implementation Details

Instruction Preprocessing. To ensure consistency and facilitate downstream analysis, each instruction was preprocessed using a transformer-based spaCy (Honnibal et al., 2020) pipeline (en_core_web_trf). This step includes tokenization, lemmatization, part-of-speech tagging, and dependency parsing. For each token in the instruction, we extracted linguistic features such as surface form, lemma, part-of-speech tag, syntactic dependency label, syntactic head, and stopword status.

Semantic Matching via Embeddings. To align mentioned arguments (actants) in the instruction with entities in the environment, we used semantic similarity in embedding space. Specifically, we employed the Sentence-BERT model (Reimers and Gurevych, 2019) (paraphrase-mpnet-base-v2) to encode both the textual mentions in the instruction and the object names in the environment. The resulting embeddings were compared using cosine similarity to identify the most relevant environmental entities for each mentioned actant.

Language Model Configuration. All experiments were conducted using GPT-3.5-Turbo, GPT-5-mini (via the OpenAI API), mistralai/Mistral-7B-Instruct-v0.2, Qwen/Qwen1.5-7B-Chat and google/gemma-2-9b-it. For GPT-3.5-Turbo and GPT-5-mini, we used a temperature of 0.3 and a maximum token limit of 400. The model was prompted using a consistent interaction format: a system message (*"You are a helpful assistant."*) followed by a single user message containing the input prompt. For other models we used ‘do_sample=True‘ and ‘top_p=0.95‘.

C.2 Prompt for Baseline Prompting

You are an ambiguity detection system.
You MUST return a JSON object with EXACTLY this structure:
"label": "Ambiguous" | "Unambiguous",
"ambiguities": [
"phrase": string,
"candidates": [string, string]
]
Rules:
- If there is ANY uncertainty, label = "Ambiguous"
- If label is "Unambiguous", ambiguities MUST be an empty list
- Use ONLY the given environment
- Do NOT add any extra keys
- Do NOT include explanations
- Return ONLY JSON
- Candidates must be short alternative interpretations, not explanations
- Do NOT describe feasibility, difficulty, or reasoning
- Candidates must be minimal noun-phrase interpretations only

C.3 Prompt for Frame Activation

Lexeme: lemma
Variants: variants_str
Environment: environment_str
Task:
Analyze each variant of the lexeme ’lemma’ in the given environment and identify the cognitive frame it activates.
For each variant:
- Identify the cognitive frame it activates.
- List the expected semantic roles (valency) for that frame.
Important:
If the only differences between variants are minor attributes like color, material, or size, and these do not affect the core function or role of the object in the environment, then all variants should activate the same cognitive frame and share the same valency.
Only distinguish frames if the environment suggests clearly different roles or functions for the variants.
Each output block must include:
1. The variant
2. The name of the activated frame (e.g., "Drinkware_Container")
3. A list of semantic roles (e.g., container, liquid, function)
CRUCIAL:
- Output one variant per block, starting with a dash and a space (‘- variant_name:‘).
- Never list multiple variants in the same block or line (no commas in variant names).
- Separate blocks clearly with a blank line.
Example:
Lexeme: soda
Variants: coke, pepsi, orange soda
Environment: coke, pepsi, orange soda, bottled water
- coke:
frame: Flavored_Soft_Drink
valency: beverage, brand, carbonation
- pepsi:
frame: Flavored_Soft_Drink
valency: beverage, brand, carbonation
- orange soda:
frame: Citrus_Drink
valency: beverage, fruit_flavor, carbonation

C.4 Prompt for Actant Extraction and Environment Matching

You are a semantic disambiguation assistant.
Here is a list of actants extracted from an instruction:
Actants: actant_text
And a list of objects present in the current environment:
Environment: environment_text
Your tasks:
1. For each actant, decide if it is ambiguous in the current environment.
2. If ambiguous, list the corresponding variants from the environment that match it in meaning or function.
3. If unambiguous, write "no variations".
Return your answer as a JSON object with actants as keys.
Example:
"cola": ["pepsi", "coke"],
"bottled water": "no variations"

D Appendix D – Additional Evaluation Results

Table 4: HR and CHR scores across instruction types on Ambik dataset.

Model	Method	Instruction type							
		Unambiguous		Common Sense		Preferences		Safety	
		HR	CHR↑	HR	CHR↑	HR	CHR↑	HR	CHR↑
GPT	LoFreeCP	0.23	0.77	0.20	0.80	0.15	0.15	0.24	0.24
	LAP	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
	KnowNo	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
	SVC	0.49	0.51	0.48	0.51	0.77	0.77	0.43	0.43
Mistral	LoFreeCP	0.72	0.28	0.69	0.31	0.73	0.73	0.62	0.62
	LAP	0.07	0.93	0.05	0.95	0.06	0.06	0.04	0.04
	KnowNo	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
	SVC	0.49	0.51	0.77	0.23	0.77	0.77	0.44	0.44
Qwen	LoFreeCP	0.64	0.36	0.74	0.26	0.62	0.62	0.17	0.17
	LAP	0.27	0.73	0.36	0.64	0.28	0.28	0.29	0.29
	KnowNo	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
	SVC	0.46	0.54	0.39	0.61	0.71	0.71	0.39	0.39
Gemma	LoFreeCP	0.49	0.51	0.56	0.44	0.20	0.20	0.34	0.34
	LAP	0.34	0.66	0.11	0.89	0.00	0.00	0.13	0.13
	KnowNo	0.00	1.00	0.00	1.00	0.00	0.00	0.00	0.00
	SVC	0.17	0.83	0.16	0.84	0.37	0.37	0.21	0.21

E Appendix E – Details of Ambiguity Detection of the SVC Method

In order to clarify the stepwise functioning of the SVC method, we now examine its implementation in detail:

Algorithm 2: Ambiguity Detection via Semantic Valency Conflicts

Input: Instruction I, environment objects $\mathcal{E}$
Output: Ambiguity type (clear/variable/conflict), semantic variants

1. Linguistic Analysis;
$\mathcal{P} \leftarrow$ extract predicates from I;
$\mathcal{A} \leftarrow$ extract actants for each $p \in \mathcal{P}$;

2. Multi-Stage Referent Matching;
foreach *actant* $a \in \mathcal{A}$ **do**
  if *exact match between* a *and* $e \in \mathcal{E}$ **then**
    $R_a \leftarrow \{e\}$ `// Single referent`
  end
  else
    $R_a \leftarrow \{e \in \mathcal{E} \mid \text{similarity}(a, e) > \tau\}$ `// Possible referents`
    if $|R_a| > 1$ **then**
      mark a as ambiguous
    end
  end
end

3. Semantic Frame Activation;
if *any actant is ambiguous* **then**
  $\mathcal{V} \leftarrow$ generate semantic variants via LLM;
  $\mathcal{F} \leftarrow$ activate frames for each variant;
end

4. Conflict Analysis;
foreach *predicate* $p \in \mathcal{P}$ **do**
  if *multiple incompatible frames for* p **then**
    report **conflict** `// Ex: "cut" → slice/chop`
  else
    if *multiple compatible frames for* p **then**
      report **variability** `// Ex: "bread" → toast/sandwich`
    else
      report **clear**
    end
  end
end

5. Intent Alignment;
if *user intent* U *provided* **then**
  compute alignment between U and $\mathcal{V}$;
end

return ambiguity type, semantic variants $\mathcal{V}$;